

Chapter 1

Using Oracle ASM

ORACLE DATABASE 11g: ADMINISTRATION II EXAM OBJECTIVES COVERED IN THIS CHAPTER:

✓ Database Architecture and ASM

- Describe Automatic Storage Management (ASM)
- Set up initialization parameter files for ASM and database instances
- Start up and shut down ASM instances
- Administer ASM diskgroups



Automatic Storage Management (ASM) provides a centralized way to manage Oracle Database disk storage. The Oracle Database 11g OCP exam will test your knowledge of ASM, and thus ASM is the first topic we will tackle in this book.

In this chapter, we will discuss what ASM is, how to configure an ASM instance, how to manage an ASM instance, and finally, how to use ASM from within an Oracle database. As is true in the rest of this book, we will conclude with a review of the chapter, and we will tell you, in the section “Exam Essentials,” what you need to know about ASM for the exam. Finally, we will present chapter review questions and answers so you can determine whether you have gained sufficient knowledge of ASM to pass the Oracle Database 11g OCP exam.

On with the show!



Exam objectives are subject to change at any time without prior notice and at Oracle’s sole discretion. Please visit Oracle’s Training and Certification website (<http://www.oracle.com/education/certification/>) for the most current exam-objectives listing.

What Is ASM?

ASM is designed to simplify Oracle database storage administration. Database environments have become more and more complex, with large numbers of (and larger) datafiles, storage area networks (SANs), and high-availability requirements. ASM is somewhat like a logical volume manager, allowing you to reduce the management of Oracle files into ASM disk groups. It also provides redundancy configurations, rebalancing operations, and, when installed on top of clusterware, the ability to share database-related files.

ASM stores files in *disk groups*, which are logical entities made up of one or more physical disk drives. ASM is good for more than just storing database datafiles. In an ASM instance, you can store database datafiles, online redo logs, archived redo logs, backup files, and data-pump dumpfiles as well as change-tracking files and control files of one or several Oracle databases, though these databases and the ASM instance must have affinity to a given machine or cluster. ASM also provides the ability to locate the flash recovery area on an ASM disk group, so your backups to disk can be made to ASM.

Here are some features of ASM:

- Automatic software data striping (RAID-0)
- Load balancing across physical disks
- Software RAID-1 data redundancy with double or triple mirrors
- Elimination of fragmentation
- Simplification of file management via support for Oracle Managed Files (OMF)
- Ease of maintenance

ASM fits perfectly into a Real Application Clusters (RAC) environment, but you can use ASM in a non-RAC environment too. In the following sections we will cover these ASM-related topics:

- The ASM instance
- Configuring ASM disks
- Accessing ASM from the database
- Managing ASM
- ASM data dictionary views

You should be aware of a few ASM limitations:

- ASM limits you to 63 disk groups in a given storage system. A disk group is a logical storage entity that is made up of one or more physical disks (we discuss adding ASM disk groups later in this chapter).
- You can have a maximum of 10,000 ASM disks in a given storage system.
- Each ASM disk can be a maximum of 4 petabytes (PB) in size.
- Each ASM instance can manage up to 40 exabytes of storage.
- Each disk group can contain up to one million files.
- Maximum file sizes vary by the type of disk group:
 - External-redundancy disk group: 140PB maximum file size
 - Normal-redundancy disk group: 42PB maximum file size
 - High-redundancy disk group: 15PB maximum file size

Working with the ASM Instance

Driving ASM is the *ASM instance*, which is a separate instance from any database instance. The ASM instance is mounted but never open like an Oracle database. It is, essentially, just a bunch of programs (daemons) running. You will create only one ASM instance per node. You can use the *Oracle Database Configuration Assistant (DBCA)* to create the ASM instance for you, or you can choose to create the ASM instance yourself.



Real World Scenario

Using ASM in the Real World

If you are an Oracle database administrator, ASM will very much be on your mind if you have to deal with Oracle database clustering (RAC). Early on in the architecting process, you will have to decide how to store shared files because RAC depends on sharing of database datafiles among the different nodes of the cluster.

You have several options, including shared raw devices and vendor-supplied products. Another option is ASM sitting on top of Oracle Clusterware. This combination can often provide a less-expensive solution over the vendor-supplied options and is easier to manage than raw disk storage. ASM and Clusterware are very popular options, and it is likely you will encounter them in your Oracle career.

One very positive thing, from a business point of view, is that ASM is included as part of your Oracle database license. This makes Oracle Clusterware/ASM an attractive alternative to products from other vendors that charge extra for their clustering solutions.

At one location we are familiar with, the move to ASM was cautious but direct. Oracle Clusterware was installed and ASM was sitting on top. All new Oracle RAC installs started using ASM instead of shared raw disk partitions. The flash recovery area (FRA) was also moved to ASM. Older RAC databases were moved to ASM over time, and now all clustered databases are using ASM.

We also note that Oracle's future direction calls for the elimination of raw devices for storage of database datafiles. This will make movement to an ASM solution even more attractive.

Creating the ASM Instance

In the following sections, we will review how to create an ASM instance. First we will cover creating the ASM instance with the Oracle Database Configuration Assistant. Then we will cover how to manually create an ASM instance.

Creating the ASM Instance with the DBCA

The DBCA is a Java-based tool that you can use to create or remove Oracle databases. Fortunately for us, it can also be used to create or remove ASM instances. You can create the ASM instance at any time, regardless of whether a database already exists. In Exercise 1.1, you will see how to create an ASM instance.

EXERCISE 1.1**Creating an ASM Instance with the DBCA.**

To create the ASM instance with the DBCA, do the following:

1. Start the Oracle DBCA.
2. The DBCA presents a list of options for you to choose from. Select Configure Automatic Storage Management and click Next.
3. The DBCA then prompts you for the SYS password for the new ASM instance to be created. Enter the password for the SYS account.
4. Oracle then creates the ASM instance. A new window appears giving you the option to create new disk groups. You can choose to create disk groups (we will cover that shortly) or you can click Finish to complete the ASM installation.
5. The name of the resulting instance will be +ASM. You can log into the ASM instance from SQL*Plus, as shown in this example:

```
C:\Documents and Settings\Robert>Set ORACLE_SID=+ASM
C:\Documents and Settings\Robert>Sqlplus sys/Robert as sysasm
SQL*Plus: Release 11.1.0.6.0 - Production on Mon Jul 14 19:55:33 2008
Copyright (c) 1982, 2007, Oracle. All rights reserved.
Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP and Data Mining options
SQL> select instance_name from v$instance;
INSTANCE_NAME
-----
+asm
```



When we logged into the ASM instance, we did so using the SYSASM role. This role is much like the SYSDBA role when logging into a database and should be used when logging into an ASM instance for administrative purposes.

Creating the ASM Instance Manually

Manual creation of an ASM instance is fairly straightforward. If you have ever manually created a database, then manually creating an ASM instance should be easy for you. To manually create an ASM instance, you would follow these steps:

1. Create directories for the ASM instance.
2. Create the instance parameter file.

3. Perform any Microsoft Windows–specific configuration.
4. Start the ASM instance.
5. Create the ASM server parameter file (spfile).

Let's look at each of these steps in a bit more detail.

Step 1: Creating Directories for the ASM Instance

An ASM instance is almost always called +ASM. An exception exists in RAC environments where the ASM instance will be called +ASM1, +ASM2, and so on. Create the admin directories for the instance using Oracle's OFA standards. In the following examples, we assume that you have defined the ORACLE_BASE parameter.

The following are examples of the commands you might issue:

For Unix

```
mkdir -p $ORACLE_BASE/admin/+ASM/bdump
mkdir -p $ORACLE_BASE/admin/+ASM/cdump
mkdir -p $ORACLE_BASE/admin/+ASM/hdump
mkdir -p $ORACLE_BASE/admin/+ASM/pfile
mkdir -p $ORACLE_BASE/admin/+ASM/udump
```

For Microsoft Windows

```
mkdir %ORACLE_BASE%\admin\+ASM\bdump
mkdir %ORACLE_BASE%\admin\+ASM\cdump
mkdir %ORACLE_BASE%\admin\+ASM\hdump
mkdir %ORACLE_BASE%\admin\+ASM\pfile
mkdir %ORACLE_BASE%\admin\+ASM\udump
```

Of course, you could use a tool such as Windows Explorer to create the directories.

Step 2: Creating the Instance Parameter File

The ASM instance will require a *parameter file*, just as any other Oracle instances does. The number of parameters that you will need to set for an ASM instance is relatively small, so the parameter file typically is smaller than that of a normal Oracle database. You will also find that some parameters that you will set are unique to ASM. Let's look at a sample parameter file, and then we will discuss ASM-specific parameters and what they are for.

First, here is an example ASM parameter file:

```
#####
# ASM Parameter File - Unix
# Note, the ASM_DISKGROUPS parameter is commented out for now.
# ASM_DISKGROUPS='DISK GROUP_ROB1'
#####
```

```
# Diagnostics and Statistics
#####
background_dump_dest=/u01/app/oracle/admin/+ASM/bdump
core_dump_dest=/u01/app/oracle/admin/+ASM/cdump
user_dump_dest=/u01/app/oracle/admin/+ASM/udump
#####
# Miscellaneous
# Of course – set compatible to your version of Oracle
#####
instance_type=asm
compatible=11.1.0.6.0
remote_login_passwordfile=exclusive
```

Note the following Oracle parameters that are specific to ASM instances:

- **INSTANCE_TYPE:** Used only with an ASM instance, this parameter indicated to Oracle that this is an ASM instance. The default value is RDBMS, which indicates the instance is an Oracle database instance. This parameter is not dynamic and is the only mandatory parameter in an ASM instance.
- **ASM_DISKSTRING:** This parameter indicates where Oracle should search for disk devices to be used by ASM. We will discuss this parameter in more detail later in this section. This parameter can be dynamically changed.
- **ASM_DISKGROUPS:** This parameter lists ASM disk groups that ASM should mount when it is started. You can also use the `alter diskgroup all mount` command to cause these disk groups to be mounted. This parameter can be dynamically changed.
- **ASM_POWER_LIMIT:** This parameter controls the rate at which ASM can rebalance disks by increasing or decreasing the degree of parallelism used. Lower values will slow rebalancing but will also result in less of an IO impact by those operations. Higher values may speed up rebalancing by parallelizing the rebalance operation. The default is 1, and this is typically sufficient. This parameter can be set dynamically.

Did you notice that we did not include any of the memory settings (for example, `SHARED_POOL_SIZE` or `DB_CACHE_SIZE`) in this parameter file? While ASM does allocate memory, the default settings for the memory parameters are often quite enough.

Step 3: Performing Any Microsoft Windows–Specific Configuration

If you are running in a Windows environment, you will need to create the ASM service with the `oradim` utility. Here is an example of this operation:

```
C:\Oracle> oradim -new -asmsid +ASM -syspwd my_password
-pfile C:\oracle\product\11.1.0.6\admin\+ASM\pfile\init.ora -spfile
-startmode manual -shutmode immediate
```

Note in this example that we made the start mode manual so the service will not start automatically when the system is started. You will want to configure the service startup as required by your system.

Step 4: Starting the ASM Instance

You are now ready to start the Oracle ASM instance. Note that until you have added a disk group, you will get an error when the ASM instance is started. This is expected. In Exercise 1.2, you will see how to start an ASM instance.

EXERCISE 1.2

Starting an ASM Instance

Starting an ASM instance is quite easy, as shown in this exercise.

1. The name of the resulting instance will be +ASM. You can log into the ASM instance from SQL*Plus, as shown in this example:

```
C:\Documents and Settings\Robert>Set ORACLE_SID=+ASM
C:\Documents and Settings\Robert>Sqlplus sys/Robert as sysasm
SQL*Plus: Release 11.1.0.6.0 - Production on Mon Jul 14 19:55:33 2008
Copyright (c) 1982, 2007, Oracle. All rights reserved.
Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP and Data Mining options
```

2. Now, start the ASM instance with the startup command:

```
SQL> startup
ASM instance started
Total System Global Area 83886080 bytes
Fixed Size 1247420 bytes
Variable Size 57472836 bytes
ASM Cache 25165824 bytes
ORA-15110: no disk groups mounted
```

You will get an ORA-15110 error, but this is no concern at this time. This error is expected because you have not yet created any ASM disk groups. We will cover the creation and management of ASM disk groups later in this chapter.

Step 5: Creating the ASM Spfile

Having started the ASM instance, create the instance spfile from the pfile created in step 2. Here is an example:

```
create spfile from pfile='/u01/opt/oracle/admin/+ASM/pfile/init.ora';
```

Managing the ASM Instance

Management of the ASM instance is typically done from the command-line prompt. In the following sections, we will discuss these topics:

- Starting and stopping the ASM instance
- ASM processes
- ASM disk discovery
- Redundancy, striping, and templates
- Adding an ASM disk group
- Dropping an ASM disk group
- Altering an ASM disk group
- Using the ASMCMD command-line utility

Starting and Stopping the ASM Instance

Starting and stopping the ASM instance is pretty straightforward and much like starting and stopping an Oracle database. Oracle knows that you are starting an ASM instance, so it knows that when you issue the `startup` command, it needs to do something a bit different from what it would with a normal database. Exercise 1.2, shown earlier in this chapter, walks you through starting an ASM instance.

Shutting down the ASM instance is just as easy. A `shutdown immediate`, `shutdown abort`, or just a plain `shutdown` will do fine. If you execute a normal or immediate `shutdown` command on an ASM instance, that shutdown will fail if there is any database using that ASM instance. An error will be returned and the ASM instance will stay up. As a result, before you shut down the ASM instance, you will need to shut down all databases using that ASM instance.

You can perform a `shutdown abort` on the ASM instance. This will cause the ASM instance to shut down immediately and all of the associated databases will be shut down in an inconsistent state. This will require instance recovery when the databases are restarted, which can increase the time it takes to reopen the database. Oracle recommends that you not use the `shutdown abort` command when stopping an ASM instance.

ASM Processes

After you start your ASM instance, you will find that several of the Oracle processes you are acquainted with will be running, such as PMON and DBWR. Additional ASM processes will be started too. These processes include the following:

- The `ARBn` process, used to perform disk group rebalance operations. There may be one or more of these processes running.
- The `ASMB` process manages ASM storage and provides statistics.
- The `GMON` process maintains disk membership in ASM disk groups.
- The `KATE` process performs proxy I/O to ASM metadata files when a disk is offlined.

- The MARK process is responsible for marking ASM allocation units as stale following a missed write to an offline disk.
- The RBAL process runs in both database and ASM instances. RBAL is responsible for performing a global open of ASM disks in normal databases. RBAL coordinates rebalance activity for disk groups in ASM instances.

ASM Disk Discovery

ASM disk discovery is the first step to setting up an ASM disk group. In this section, we will cover configuring the `ASM_DISKSTRING` parameter, which helps with ASM disk discovery, and then we will discuss the topic of ASM disk discovery in general.

Setting the *ASM_DISKSTRING* Parameter

When you configured the parameter file for your ASM instance, you configured a parameter called `ASM_DISKSTRING`. This parameter contains the paths that Oracle will use to try to find the various candidate disks available for ASM's use. The process of ASM finding disks in the `ASM_DISKSTRING` path is known as *discovery*.

You may not need to set `ASM_DISKSTRING`. `ASM_DISKSTRING` has a number of different default values depending on the platform you are using. Table 1.1 lists the platform-specific default values (these will be set if `ASM_DISKSTRING` is set to a NULL value only).

TABLE 1.1 Default ASM Disk String

Platform Name	Default ASM_DISKSTRING Value
AIX	/dev/rhdisk*
HP-UX	/dev/rdsk/*
Linux	/dev/raw/*
Mac OS X	/dev/rdisk*s*s1
Solaris	/dev/rdsk/*
Tru64UNIX	/dev/rdisk/*

You can have multiple locations in the `ASM_DISKSTRING` parameter (we will provide an example of this in just a moment). If you insert a `?` placeholder at the beginning of the string, Oracle will expand that out to represent the location of `ORACLE_HOME` in the parameter values. The `ASM_DISKSTRING` can be dynamically altered, which is nice if your friendly system administrator adds some storage to your system that you want Oracle to be able to use. If you happen to change `ASM_DISKSTRING` dynamically and the new disk path is not present, it will revert to the old disk path. Removing an existing disk path, when that disk path is in use, will result in a failure of the command.

Another thing to consider when determining how to configure the `ASM_DISKSTRING` parameter is performance. Leaving this parameter set to `NULL`, and thus taking the Oracle default, will often be sufficient. However, if you set `ASM_DISKSTRING` using a more restrictive set of parameters, you may find that discovery of disks will be faster. For example, using the default Linux setting of `/dev/raw/*` will result in ASM scanning the entire `/dev/raw` file system structure (it does not search subfolders). If you have a large number of devices in this structure, this may take some time. If, however, your disk devices in this structure are all prefixed with the word *raw* (`raw1`, `raw2`, `raw3`, and so on), then setting the `ASM_DISKSTRING` to `/dev/raw/raw*` could reduce the time it takes ASM to perform discovery and improve performance of the startup of the ASM instance.

Something you will see common to all `ASM_DISKSTRING` parameters is the use of the asterisk. The asterisk is required when defining the `ASM_DISKSTRING` parameter. Here are some examples of setting the `ASM_DISKSTRING` parameter. In this first example, ASM will look for disks in devices when we create disk groups:

```
Alter system set ASM_DISKSTRING='/devices/*';
```

In the next example, we are pointing `ASM_DISKSTRING` to `ORACLE_HOME/disks`:

```
Alter system set ASM_DISKSTRING='?/disks/*';
```

In this example, we are pointing `ASM_DISKSTRING` to two different locations:

```
Alter system set ASM_DISKSTRING='?/disks/d1/*,?/disks/d21/*';
```

We could also use some adjunctive regular expressionish-type extensions and perform the allocation this way:

```
Alter system set ASM_DISKSTRING='?/disks/d[12]/*';
```

ASM Disk Discovery on Instance Start

When the ASM instance is started, it will use the paths listed in the `ASM_DISKSTRING` parameter and discover the disks that are available. These disks can then be added to ASM disk groups that we will discuss in the next section. Once discovery is complete and the ASM instance is open, you can review the disks discovered by looking at the `V$ASM_DISK` view, as shown in this example:

```
column path format a20
set lines 132
set pages 50
select path, group_number group_#, disk_number disk_#, mount_status,
header_status, state, total_mb, free_mb
from v$asm_disk
order by group_number;
```

PATH	GROUP_#	DISK_#	MOUNT_S	HEADER_STATU	STATE	TOTAL_MB	FREE_MB
/dev/raw/raw4	0	1	CLOSED	FOREIGN	NORMAL	39	0
/dev/raw/raw5	0	0	CLOSED	FOREIGN	NORMAL	39	0
/dev/raw/raw3	0	2	CLOSED	FOREIGN	NORMAL	39	0
/dev/raw/raw6	0	2	CLOSED	CANIDATE	NORMAL	2048	2048
ORCL:ASM01_004	1	3	CACHED	MEMBER	NORMAL	34212	30436
ORCL:ASM01_005	1	4	CACHED	MEMBER	NORMAL	34212	30408
ORCL:ASM01_006	1	5	CACHED	MEMBER	NORMAL	34212	30420
ORCL:ASM01_007	1	6	CACHED	MEMBER	NORMAL	34212	30297
ORCL:ASM01_008	1	7	CACHED	MEMBER	NORMAL	34212	30507
ORCL:ASM01_009	1	8	CACHED	MEMBER	NORMAL	34212	30404
ORCL:ASM01_010	1	9	CACHED	MEMBER	NORMAL	34212	30509
ORCL:ASM01_011	1	10	CACHED	MEMBER	NORMAL	34212	30449
ORCL:ASM01_012	1	11	CACHED	MEMBER	NORMAL	34212	30340
ORCL:ASM01_013	1	12	CACHED	MEMBER	NORMAL	34212	30357

In this view, you see that there are three disks that are not assigned to any group (those with GROUP_# set to 0). These are unassigned disks that ASM has discovered but that have not been assigned to a disk group. Note the mount status of CLOSED on those three disks, which also indicates that the disk is not being accessed by ASM. The HEADER_STATUS of FOREIGN indicates that these disks contain data already and are owned by some process other than ASM (in this case, these are voting disks for a RAC). If the HEADER_STATUS says CANIDATE, as with /dev/raw/raw6, then we could add this disk to an ASM disk group.

Notice that most of the disks have a MOUNT_STATUS of CACHED and a HEADER_STATUS of MEMBER. This means that the disk is currently part of an ASM disk group (which we will discuss more in the next section).

There are some cases where the V\$ASM_DISK view will not report any disks. For example, on our Windows XP system there are no raw disks to discover, so the V\$ASM_DISK view will simply be blank. This is not a problem because we can use an existing file system as a location for an ASM disk. We will discuss that in the next section as we show you how to add disk groups to ASM.

Here are some things to be aware of with regard to ASM disk discovery:

- ASM can discover no more than 10,000 disks. If you have more than that, ASM will discover only the first 10,000 disks. This can occur when your ASM disk string is not sufficiently restrictive and the directory that you are searching in has a number of raw devices but many of them are not going to be assigned to ASM.
- ASM will not discover any disk that contains an operating-system partition table.
- ASM may discover disks that already contain Oracle data (as in our previous example with the voting disks).

Redundancy, Striping, and Other ASM Topics

When configuring ASM disk groups, you need to consider recoverability, performance, and other attributes. We will first cover recoverability by discussing the concept of redundancy. We will then discuss striping of ASM disk groups and ASM disk templates. Then we'll discuss ASM disk group attributes, ASM fast disk resync features, and ASM preferred mirror read features. We will end this section with a discussion of ASM Allocation Unit (AU) size and extents in ASM.

Redundancy

When configuring an ASM disk group, you can use one of three different *ASM redundancy* setting options to protect the data in your disk group:

- *Normal*: Typically employs two-way mirroring by default and thus requires allocation of two failure groups.
- *High*: Typically employs three-way mirroring by default and thus requires allocation of three failure groups.
- *External*: Does not employ any mirroring. This setting is typically used when the disk group is being assigned to an external disk that is attached to some device that already employs some disk redundancy.

The Costs of Redundancy

Keep in mind that there is a cost to everything, and this includes redundancy. If you have two 100GB ASM disks that you will be assigning to an ASM disk group, you will be able to effectively use only 100GB of overall space if you use normal redundancy, because each disk will have to go into an individual failure group. If you were to use external redundancy, you would be able to use all 200GB (at a cost, of course, of loss of protection).

Redundancy is supported by one or more *failgroups* (or failure groups) assigned to the ASM disk group when it is created. If you are using external redundancy, you typically would just have one failure group. If you are using the Normal redundancy setting, then the ASM disk group typically will need two failure groups. Each failure group represents a logical allocation of one or more disks to the ASM disk group and provides for mirroring within that disk group. Thus, when you create an ASM disk group, you might have one disk assigned to failure group 1 and one disk assigned to failure group 2. This way your data is protected from failure.

When you're using ASM mirroring, ASM will allocate an extent on a disk that becomes the primary copy (one of the failure groups) and then allocate copies of that extent to the mirrored copies (the other failure groups). When you create a disk group, you can indicate which disk goes in which failure group or you can let Oracle decide for you.

When you define the redundancy setting for a disk group, you are defining things such as what kind of striping occurs and whether the data will be mirrored. These attributes are defined based on which template you have assigned to the ASM disk group. By default, when

you create a disk group, Oracle will assign it the default template setting. You can optionally assign another ASM template to a given disk group (We discuss templates later in this chapter).

Table 1.2 gives you some guidance about the redundancy-related settings defined within the default template.

TABLE 1.2 Default-Template Redundancy Settings

Template Name	Striping	Mirroring with Normal Redundancy	Mirroring with High Redundancy	Mirroring with Extended Redundancy
Control file	Fine	Three-way mirroring	Three-way mirroring	No mirroring
Datafile	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Online log	Fine	Two-way mirroring	Three-way mirroring	No mirroring
Archivelog	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Tempfile	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Backupset	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Parameterfile	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Data guard config	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Flashback	Fine	Two-way mirroring	Three-way mirroring	No mirroring
Change tracking	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Dumpset	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Xtransport	Coarse	Two-way mirroring	Three-way mirroring	No mirroring
Autobackup	Coarse	Two-way mirroring	Three-way mirroring	No mirroring

Default ASM Template Redundancy Settings

So, if you create a disk group with normal redundancy using the default template and you put datafiles on it, the datafile template would be used by default. In this case, a datafile would use two-way mirroring and coarse striping (see the section “Striping”). This means you would have to allocate at least two disks to an ASM disk group when it was created, one assigned to a different failure group. We will discuss failure groups later in this chapter.

Dealing with ASM Disk Loss

If you lose an ASM disk, then one of two situations will occur. First, ASM will take the lost/damaged disk offline and then automatically drop it. ASM will attempt a rebalance operation to maintain redundancy, using the mirror copies as required. The disk group and its associated data will remain available during this time.

If the disk group cannot be rebalanced, then ASM will take the whole disk group offline and the data in that disk group will not be available until the damaged disk is restored and the disks can be rebalanced.

Striping

Table 1.2 includes a *striping* column. There are two values there, fine and coarse. This refers to the stripe size that ASM applies to the disks that the disk groups are assigned to. If fine striping is selected, the ASM will use a 128KB stripe size. If coarse is selected, then Oracle uses the AU size of the disk group for the stripe size.

Templates

When you create an ASM disk group, Oracle will assign a default *template* to that disk group (see Table 1.2). A template is simply a named collection of attributes. For example, if you create a disk group using the default template and then create datafiles in that disk group, the datafile template will define the redundancy and striping for that data.

There may be cases where you want to define your own template for a disk group. You will need to first create the disk group and then alter it using the `add template` parameter of the `alter diskgroup` commands, as shown in this example:

```
CREATE DISKGROUP sp_dgroup2 NORMAL REDUNDANCY
failgroup diskcontrol1 DISK 'c:\oracle\asm_disk\_file_disk3' NAME file_diska1
failgroup diskcontrol2 DISK 'c:\oracle\asm_disk\_file_disk4' NAME file_diskb1;
ALTER DISKGROUP sp_dgroup2 ADD TEMPLATE new_template ATTRIBUTES (mirror);
```

After the mirror template has been added, you can create files in that disk group using the new template. When you add a template to a disk group, the template cannot be retroactively applied to files already in that disk group. As a result, you will need to use RMAN to back up and then restore files that already exist in the disk group in order for them to take on the attributes of the new template.

You can see the templates associated with a given disk group by querying the V\$ASM_TEMPLATE view, as shown in this example:

```
SQL> select * from v$asm_template
2 where group_number=2;
GROUP_NUMBER ENTRY_NUMBER REDUND STRIPE S NAME
-----
```

2	0	MIRROR	COARSE	Y	PARAMETERFILE
2	1	MIRROR	COARSE	Y	DUMPSET
2	2	HIGH	FINE	Y	CONTROLFILE
2	3	MIRROR	COARSE	Y	ARCHIVELOG
2	4	MIRROR	FINE	Y	ONLINELOG
2	5	MIRROR	COARSE	Y	DATA FILE
2	6	MIRROR	COARSE	Y	TEMPFILE
2	7	MIRROR	COARSE	Y	BACKUPSET
2	8	MIRROR	COARSE	Y	AUTOBACKUP
2	9	MIRROR	COARSE	Y	XTRANSPORT
2	10	MIRROR	COARSE	Y	CHANGETRACKING
2	11	MIRROR	FINE	Y	FLASHBACK
2	12	MIRROR	COARSE	Y	DATAGUARDCONFIG
2	13	MIRROR	COARSE	N	NEW_TEMPLATE

In this output, you can see that our new template (`new_template`) has been created and is ready for use. You can drop a template with the `alter diskgroup` command using the `drop template` parameter, as shown in this example:

```
ALTER DISKGROUP sp_dgroup2
DROP TEMPLATE new_template;
```

And you can alter a user-defined template with the `alter template` parameter of the `alter diskgroup` command. Notice in this example that we are actually changing one of the attributes of the default templates. You cannot drop the default templates, but you can modify them, as shown here:

```
ALTER DISKGROUP sp_dgroup2
ALTER TEMPLATE datafile
ATTRIBUTES (coarse);
```

ASM Disk Group Attributes

We have discussed ASM templates that define a set of attributes to the disk group assigned to them. Oracle Database 11g also allows you to define specific *disk group attributes*. Disk

group attributes are set using the `attribute` clause of the `create diskgroup` and `alter diskgroup` commands. The following attributes can be set on a specific ASM disk group:

Au_size This is the disk group allocation unit (AU) size. The value defaults to 1MB and can be set only when the disk group is created. You must modify the AU size of the disk group if you want the disk group to be able to hold larger amounts of data. A disk group with the default AU size will be able to grow to 35TB (normal redundancy). Increasing the AU size will significantly increase the maximum size of the disk group. The maximum AU size is 64MB.

Compatible.rdbms Indicates the database version that the disk group is compatible with at a minimum (default is 10.1). This value should be equal to or greater than the compatibility parameter of the database(s) accessing the ASM disk group. This value cannot be rolled back once set.

Compatible.asm Indicates the ASM instance version that the disk group is compatible with at a minimum (default is 10.1). **Compatible.asm** must always be set to a value equal to or greater than **compatible.rdbms**. Once **compatible.asm** is set for a disk group, it can not be rolled back to an earlier value.

Disk_repair_time Indicates the length of time that the disk resync process should maintain change tracking before dropping an offline disk. The default for this parameter is 3.6 hours.

Disk group attributes can be viewed using the `V$ASM_ATTRIBUTE` view. You can see some examples of setting compatibility here:

```
Create diskgroup robert01 external redundancy
Disk '/oracle/asm/ASM_DISKGROUP_robert01.asm'
Attribute 'compatible.asm'='11.1.0';
Alter diskgroup robert01 set attribute 'DISK_REPAIR_TIME'='1200M';
Alter diskgroup robert01 set attribute 'compatible.asm'='11.1.0';
```

ASM Fast Disk Resync

The redundancy features of ASM make it possible for an ASM disk group to survive the loss of a disk associated with that disk group. Disk loss can result from a number of reasons, such as loss of controller cards, cable failures, or power-supply errors. In many cases, the disk itself is still intact. To allow for sufficient time to recover from disk failures that do not involve the actual failure of a disk, ASM provides the *ASM fast disk resync* feature.

By default, when a disk in an ASM disk group fails (including any associated infrastructure pieces), the disk will be taken offline automatically. The disk will be dropped some 3.6 hours later. As a result, you have only 3.6 hours by default to respond to a disk outage. If you correct the problem and the physical disk media is not corrupted, then ASM fast disk resync will quickly resynchronize the disk when it comes back online, correcting the problem very quickly. This type of resynchronization is much faster than rebuilding a newly added disk should the disk media be corrupted.

You can change the amount of time that Oracle will wait to automatically drop the disk by setting the `disk_repair_time` attribute (see the discussion on attributes earlier, in the section “ASM Disk group Attributes”) for the individual disk groups using the `alter diskgroup` command, as shown in this example, where we set the `disk_repair_time` attribute to 18 hours:

```
Alter diskgroup dgroup1 set attribute 'disk_repair_time'='18h';
```

ASM Preferred Mirror Read

The *ASM preferred mirror read* feature allows you to define a primary set of disks that are the preferred disks to read from for a given instance. This is most prevalent when using RAC databases. In a RAC configuration, you could have two or more sets of disk arrays. Each disk array might be local to a given RAC instance. ASM preferred mirror read allows you to indicate which disk array is local to a specific RAC instance. As a result, it becomes the preferred disk set for the instance and thus is likely to be more performant.

The ASM preferred local disk is defined using the optional parameter `asm_preferred_read_failure_groups`.

ASM AU Size and Extents

ASM files are stored in disk groups. In each disk group, space is allocated in *extents*, and an extent consists of one or more units of space called *allocation units (AUs)*. Allocation units default to a size of 10MB and can be configured from 1 to 64MB at the time the disk group is created. Once the AU size has been determined for a given disk group, it cannot be changed.

To enable support for larger ASM datafiles, to reduce the memory overhead of large databases, and to improve file open and close operations, ASM uses a variable-extent sizing policy. Extents will be initially sized at the size of the AU (for the first 20,000 extents) of the ASM disk group in which the extent is created. The extent size will be increased to 8 times the AU size for the next 20,000 extents and then will increment to 64 times the AU size for subsequent extent allocations.

Adding an ASM Disk Group

We have now talked about discovering disks, and we have talked about templates, redundancy, and striping. Now we need to talk about actually creating a disk group. You use the `create diskgroup` command to create an ASM disk group. When you issue the command, you will assign the disk group its name, and you will add one or more discovered (unallocated) disks to that disk group. Here is an example of the use of the `create diskgroup` command:

```
CREATE DISKGROUP dgroup1 NORMAL REDUNDANCY
failgroup diskcontrol1 DISK
'/devices/diska1'
failgroup diskcontrol2 DISK
'/devices/diskb1';
```

In this case, we have created a disk group called `dgroup1`. It is using normal redundancy and the default template. Two named failure groups are assigned, `diskcontrol1` and `diskcontrol2`. Each failure group represents one physical or logical disk unit, which has been discovered by ASM. Two separate disks and failure groups are required because of the normal redundancy. If we used high redundancy, we would need to add a third disk to the command, as shown here:

```
CREATE DISKGROUP dgroup1 HIGH REDUNDANCY
failgroup diskcontrol1 DISK
'/devices/diska1' NAME diska1
failgroup diskcontrol2 DISK
'/devices/diskb1' NAME diskb1
failgroup diskcontrol3 DISK
'/devices/diskc1' NAME diskc1;
```

You might have noticed the `name` clause in the `create diskgroup` command example earlier. You can also name the disks being assigned to the ASM disk group using the `name` clause of the `create diskgroup` command. Failure to use the `name` clause will result in each disk receiving its own system-default assigned name.

When you create an ASM disk group, Oracle will add that disk group to the `ASM_DISKGROUPS` parameter on the ASM instance only if you are using an `spfile`. If you are not using an `spfile`, you will need to manually add the disk group to the `ASM_DISKGROUPS` parameter. The `ASM_DISKGROUPS` parameter tells Oracle which disk groups it should mount when the ASM instance is started. You can see the `ASM_DISKGROUPS` parameter setting by using the `show parameter` command from `SQL*Plus`, as shown here:

```
SQL> show parameter ASM_DISKGROUPS
```

NAME	TYPE	VALUE
ASM_DISKGROUPS	string	COOKED_DGROUP1, SP_DGROUP2

If you do not add the disk group to the `ASM_DISKGROUPS` parameter, you will need to manually mount the disk group.

You might have noticed that each time we create a new disk group (and when we add new disks to a disk group), we give the disk a name. For example, here we create a new disk group called `DGROUP1`:

```
CREATE DISKGROUP dgroup1 EXTERNAL REDUNDANCY
failgroup diskcontrol1 DISK
'/oracle01/oradata/asm/disk_group1.dsk' NAME dgroup1_0000;
```

You can reference the disk group and the disk name by joining the V\$ASM_DISK and V\$ASM_DISKGROUP views, as shown in this query:

```
select dg.name dg_name, ad.name fg_name, path
from v$asm_disk ad
right outer join v$ASM_DISKGROUP adg
on ad.group_number=adg.group_number
where adg.name='DGROUP1';
DG_NAME  FG_NAME          PATH
-----
DGROUP1  DGROUP1_0000      /oracle01/oradata/asm/disk_group1.dsk
```



Real World Scenario

Why Tiered Storage?

Tiered database storage attempts to reduce the overall costs of disk storage in databases. When you configure ASM, you might want to consider configuring different disk groups for different kinds of tiered storage.

For example, suppose you work at Amalgamated General Consolidated. You have a new database that you are designing. You can choose from fast and expensive solid-state disks that costs \$50 a gigabyte. Then there are the Fibre Channel disks that are a bit slower but only \$30 a gigabyte. Finally, there are the slow SATA drives at \$20 a gigabyte.

You can, of course, architect your 1 terabyte database with all solid-state disks at a cost of \$50 million. It will be fast, no doubt, but is this the best choice for Amalgamated?

Amalgamated decides to analyze the needs of the database and determine if it can benefit from a tiered storage approach. If you determined that you need only 100GB of solid-state disk and that you could store the remaining 900GB on your \$20 SATA drives, that would be cost savings of \$27 million. That's quite a big savings.

This type of architecture requires that you architect physical database objects to sit across these layers using partitioning, creating the more heavily used partitions on better-performing disk; the infrequently used partitions can be moved over time to the cheaper/slower disks. This might take more time and effort, but in the end the dollar savings can be significant! Tiered storage is an idea that is very much taking off!

Dropping an ASM Disk Group

To remove an ASM disk group, you use the `drop diskgroup` command. By default, if any files exist in the disk group, ASM will not allow you to drop it unless you use the `including`

contents clause. The `drop diskgroup` statement is synchronous in nature, so once the prompt returns, the deed is done ... no Recycle Bin here. When the `drop diskgroup` command is executed, ASM will unmount the disk from the ASM instance and write over all the ASM-related information on that disk. The `ASM_DISKGROUPS` parameter will also be changed if you are using an spfile. Here is an example of removing an ASM disk group with the `drop diskgroup` command:

```
Drop diskgroup sp_dgroup2;
```

If the ASM disk group has files in it, use this version:

```
Drop diskgroup sp_dgroup2 including contents;
```

Altering an ASM Disk Group

The `alter diskgroup` command is used to modify ASM disk groups. With the `alter diskgroup` command, you can do the following:

- Add disks to an ASM disk group
- Remove disks from an ASM disk group
- Add and drop disks from an ASM disk group
- Undrop disks from an ASM disk group
- Resize disks in a disk group
- Manually rebalance a disk group
- Mount and unmount disk groups
- Check the consistency of a disk group
- Create ASM disk group directories
- Manage ASM disk group directories

Adding Disks to an ASM Disk Group

As databases grow, you need to add disk space. The `alter diskgroup` command allows you to add disks to a given disk group to increase the amount of space available. Adding a disk to an existing disk group is easy with the `alter diskgroup` command, as shown in this example:

```
alter diskgroup cooked_dgroup1
add disk 'c:\oracle\asm_disk\_file_disk3'
name new_disk;
```

When you add a disk to a disk group, Oracle will start to rebalance the load on that disk group. Also, notice that in the preceding example we did not assign the disk to a

specific failure group. As a result, each disk will be assigned to its own failure group when it's created. For example, when we added the disk to the `cooked_dgroup1` disk group, a new failure group called `cooked_dgroup1_0002` was created, as shown in this output:

```
SQL> select disk_number, group_number, failgroup from v$asm_disk;
```

```
DISK_NUMBER GROUP_NUMBER failgroup
-----
1            0
0            1 DISKCONTROL1
1            1 DISKCONTROL2
2            1 COOKED_DGROUP1_0002
```

We can add a disk to an existing failure group by using the `failgroup` parameter, as shown in this example:

```
alter diskgroup cooked_dgroup1
add failgroup DISKCONTROL1
disk 'c:\oracle\asm_disk\_file_disk4'
name new_disk;
```

Removing Disks from an ASM Disk Group

The `alter diskgroup` command allows you to remove disks from an ASM disk group using the `drop disk` parameter. ASM will first rebalance the data on the disks to be dropped, assuming enough space is available. If insufficient space is available to move the data from the disk to be dropped to another disk, then an error will be raised. You can use the `force` parameter to force ASM to drop the disk, but this can result in data loss. Here is an example of dropping a disk from a disk group:

```
alter diskgroup cooked_dgroup1
drop disk 'c:\oracle\asm_disk\_file_disk4';
```

The `alter diskgroup` command also gives you the option to drop from a failure group all disks that are assigned to the disk group. Use the `in failgroup` keyword and then indicate the name of the failure group, as shown in this example:

```
alter diskgroup cooked_dgroup1
drop disks in failgroup diskcontrol1;
```

When you drop a disk from a disk group, the operation is asynchronous. Therefore, when the SQL prompt returns, this does not indicate that the operation has completed. To

determine if the operation has completed, you will need to review the V\$ASM_DISK view. When the disk drop is complete the column HEADER_STATUS will take on the value of FORMER, as shown in this example:

```
SQL> select disk_number, header_status from v$asm_disk;
DISK_NUMBER HEADER_STATU
-----
0 FORMER
1 FORMER
1 MEMBER
2 MEMBER
```

If the drop is not complete (the V\$ASM_DISK column STATE will read dropping), you can check the V\$ASM_OPERATION view and it will give you an idea of how long the operation is expected to take before it is complete. Here is an example query that will provide you with this information:

```
select group_number, operation, state, power, est_minutes
from v$asm_operation;
```

Adding and Dropping Disks from an ASM Disk Group

The alter diskgroup command will allow you to add and drop a disk from a disk group at the same time. Assuming you want to add a disk /dev/raw/raw6 and drop a disk called d2c, you could issue this command:

```
alter diskgroup mydisk group
add failgroup fg4 disk '/dev/raw/raw6/' name d2d
drop disk d2c;
```

Undropping Disks from an ASM Disk Group

You know you are having a bad day when you accidentally drop a disk from a disk group and you realize your mistake only after the drop operation has completed. Fortunately, this is not one of those mistakes that you cannot recover from. If you have accidentally dropped a disk, simply use the alter diskgroup command with the undrop disks parameter, as shown here:

```
alter diskgroup sp_dgroup2 undrop disks;
```

This will cancel the pending drop of disks from that disk group. You can not use this command to restore disks dropped if you dropped the entire disk group with the drop diskgroup command.

Resizing Disks in an ASM Disk Group

Sometimes when more space is needed, all a disk administrator needs to do is add that additional space to the disk devices that are being presented for ASM to use. If this is the case, you will want to indicate to ASM that it needs to update its metadata to represent the correct size of the disks it's using so you get the benefit of the additional space. This is accomplished using the `alter diskgroup` command with the `resize all` parameter, as shown in this example:

```
alter diskgroup cooked_dgroup1 resize all;
```

This command will query the operating system for the current size of all of the disk devices attached to the disk group and will automatically resize all disks in that disk group accordingly. You can indicate that a specific disk needs to be resized by including the disk name (from the `NAME` column in `V$ASM_DISK`), as shown in this example:

```
alter diskgroup cooked_dgroup1 resize disk FILE_DISK1;
```

You can also resize an entire failure group at one time:

```
alter diskgroup cooked_dgroup1 resize disks in failgroup DISKCONTROL2;
```

Manually Rebalancing Disks Assigned to an ASM Disk Group

Manually rebalancing disks within ASM is typically not required since ASM will perform this operation automatically. However, in cases where you might want to have some more granular control over the disk-rebalance process, you can use the `alter diskgroup` command along with the `rebalance` parameter to manually rebalance ASM disks.

When we discuss rebalancing disks in ASM, we often discuss the power level that is assigned to that rebalance operation. Setting power with regard to a rebalance operation really defines the urgency of that operation with respect to other operations occurring on the system (for example, other databases or applications). When a rebalance operation occurs with a low power (for example, 1, the typical default), then that operation is not given a high priority on the system. As a result, the rebalance operation can take some time. When a higher power setting is used (for example, 11, the maximum), the ASM is given higher priority. This can have an impact on other operations on the system. If you use a power of 0, this will have the effect of suspending the rebalance operation. You can set the default power limit for the ASM instance by changing the `asm_power_limit` parameter.

Here is an example of starting a manual rebalance of a disk group:

```
alter diskgroup cooked_dgroup1 rebalance power 5 wait;
```

In this example, you will notice that we used the `wait` parameter. This makes this rebalance operation synchronous for our session. Thus, when the SQL prompt returns, we know that

the rebalance operation has completed. The default is `nowait`, which will cause the operation to be synchronous in nature. You can check the status of the rebalance operation using the `V$ASM_OPERATION` view during asynchronous rebalance operations. If you use the `wait` parameter and you want to convert the operation to an asynchronous operation, you can simply press `Ctrl+C` on most platforms and an error will be returned along with the SQL prompt. The rebalance operation will continue, however.

If you do not use the `power` parameter during a manual rebalance operation, or if an implicit rebalance operation is occurring (because you are dropping a disk, for example), you can affect the power of that rebalance operation by dynamically changing the `ASM_POWER_LIMIT` parameter to a higher value with the `alter system` command.

Finally, you can also use the `rebalance` parameter along with the `power` parameter when adding, dropping, or resizing disks within a disk group, as shown in this example:

```
alter diskgroup cooked_dgroup1 resize all rebalance power 5;
```

Manually Mounting and Unmounting an ASM Disk Group

If an ASM disk group is not assigned to the `ASM_DISKGROUPS` parameter, or if the disk group is unmounted for some other reason, you will need to mount the ASM disk group. You can use the `alter diskgroup` command with the `mount` clause to mount the disk group.

Additionally, if you need to dismount an ASM disk group, you can use the `alter diskgroup` command. Here are some examples:

```
alter diskgroup sp_dgroup2 dismount;  
alter diskgroup sp_dgroup2 mount;
```

Note that when you dismount a disk group, that disk group will be automatically removed from the `ASM_DISKGROUPS` parameter if you are using an `spfile`. This means that when ASM is restarted, that disk group will not be remounted. If you are using a regular text parameter file, you will need to remove the disk group manually (assuming it's in the parameter file to begin with) or ASM will try to remount the disk group when the system is restarted.

Checking the Consistency of a Disk Group

On occasion you might wonder if there is some problem with an ASM disk group, and you will want to check the consistency of the ASM disk group metadata. This need might arise because of an error that occurs when the ASM instance is started or as the result of an Oracle database error that might be caused by some ASM corruption. To perform this check, simply use the `alter diskgroup` command with the `check all` parameter, as shown in this example:

```
alter diskgroup sp_dgroup2 check all;
```

When you execute the `alter diskgroup check all` command the results are written to the alert log of the instance. ASM will attempt to correct any errors that are detected.

Creating ASM Disk Group Directories

When you create an ASM disk group, it includes a system-generated directory structure for the ASM files that will be stored in that disk group. The system-generated directory structure takes on the following format, where `disk_group` is the root of the directory hierarchy:

`+disk_group/database_name/object_type/ASM_file_name`

The database name will be the name of the database that the data is associated with. The `object_type` is the type of object being stored (for example, datafile) and the `ASM_file_name` is the system-generated filename assigned to that ASM file.

ASM allows you to create your own directories within these predefined structures. This allows you to give alias names to the ASM files that you will create. This can make working with ASM files easier.

To create a directory structure, you use the `alter diskgroup` command with the `add directory` parameter, as shown in this example:

```
ALTER DISKGROUP cooked_dgroup1
ADD DIRECTORY '+cooked_dgroup1/stuff';
```

Managing ASM Disk Group Directories

The `alter diskgroup` command is also used to manage ASM disk group directories. If you want to drop an ASM disk group directory, simply use the `drop directory` clause of the `alter diskgroup` command, as shown in this example:

```
alter diskgroup cooked_dgroup1
drop directory '+cooked_dgroup1/stuff';
```

You can also rename directories with the `alter diskgroup` command using the `rename directory` clause, as shown in this example:

```
alter diskgroup cooked_dgroup1
rename directory '+cooked_dgroup1/stuff' to '+cooked_dgroup1/badstuff';
```

You can see the ASM disk group directories in the `V$ASM_ALIAS` view, as shown in this example:

```
select a.name "Alias Name", b.name "Disk group"
from v$asm_alias a, v$ASM_DISKGROUPS b
where a.group_number=b.group_number;
```

Alias Name	Disk group
-----	-----
badstuff	COOKED_DGROUP1

Using the ASMCMD Command-Line Utility

The ASMCMD tool is a command-line utility that allows you to manage ASM instances and the disk structures and files within those instances. With ASMCMD, you can do the following:

- List contents of ASM disk groups
- Perform searches (like directory listings)
- Add or remove directories
- Display space availability and utilization

ASMCMD allows you to traverse the ASM disks as you would a directory structure. From the root of the ASM instance, you can move down the various disk structures to find the disks you are interested in. Many of the commands in ASMCMD are Unix-like (in other words, `ls`, `cd`) and therefore the ASMCMD is fairly easy to learn how to use.

Starting ASMCMD

To start ASMCMD, simply set your `ORACLE_SID` to `+ASM` and then type **asmcmd** from the command line, as shown here:

```
C:\>set ORACLE_SID=+ASM
```

```
C:\>asmcmd
```

Or from Unix:

```
/opt/oracle>export ORACLE_SID=+ASM
```

```
/opt/oracle>asmcmd
```



You will need to make sure that `perl.exe` is in the path before you run ASMCMD. If you have installed more than one `ORACLE_HOME`, it may take some setup to get the pathing set correctly. Make sure the following is set to the correct `ORACLE_HOME`:

- `ORACLE_HOME`
- `PATH`
- `PERL5LIB`
- `PERLBIN`

When ASMCMD starts, you will see the ASMCMD prompt, as shown here:

```
ASMCMD>
```

You can start ASMCMD with the `-p` option and it will display the current directory level, as shown in this example:

```
C:\oracle\product\11.1.0.6\DB01\BIN>asmcmd -p
ASMCMD [+] >
```

ASMCMD Commands

ASMCMD has a basic set of commands, many of which mimic Unix commands. You can see these commands from the ASMCMD prompt if you type in **help**. The commands are pretty straightforward and easy to use. In the next section, we will introduce each command and provide an example of its use. Table 1.3 lists the different ASMCMD commands and their purposes.

TABLE 1.3 ASMCMD Commands

Command	Purpose	Example
cd	Changes ASM directory.	cd +group1
du	Gets disk use.	du
find	Finds directory or file.	find + rob11g
help	Displays the help screen.	help
ls	Lists files in directory.	ls -l
lsct	Lists all clients using the ASM instance.	lsct
lsdg	Lists information on disk groups in the ASM instance.	lsdg
lsdsk	Lists ASM visible disks. Supported only in Unix.	lsdsk -k -d DATA *
mkalias	Creates an ASM alias for a given ASM filename.	mkalias +cooked_dgroup1/11gDB/controlfile/Current.258.613087119 +cooked_dgroup1/control01.ctl
mkdir	Creates an ASM directory.	mkdir old
md_backup	Backs up ASM metadata.	md_backup -b /tmp/dgbackup070222 -g dgroup1 -g dgroup2

TABLE 1.3 ASMCMD Commands (*continued*)

Command	Purpose	Example
md_restore	Restores ASM metadata.	md_restore -t full -g dgroup1 -i /tmp/dgbackup070222
pwd	Locates where you are on the ASM directory tree.	pwd
remap	Remaps a range of physical blocks on disk.	remap data data_0003 6000-8000
rm	Removes an ASM directory or file.	rm Current.258.613087119 rm current* rm -r current*
rmalias	Removes an ASM alias.	rmalias +cooked_dgroup1/11gDB/ datafile/alias_tbs_01.dbf

Overview of ASM Data Dictionary Views

You were introduced to Oracle's data dictionary as a part of your OCA studies. Recall that the data dictionary provides views on the operation and performance of the database. The data dictionary also provides a great deal of metadata about database structures. Several data-dictionary views are available for use with ASM. Table 1.4 lists those views and gives descriptions of their use.

You can learn more about the Oracle data dictionary in Chapter 2, "Performing Oracle User-Managed Backups."

TABLE 1.4 ASMCMD Commands

View Name	In ASM Instance	In Database
V\$ASM_DISKGROUP	This view will describe a given disk group.	This view contains a single row for each ASM disk group that is mounted by the local ASM instance. Note that discovery will occur each time you query this view. This can have performance impacts.

TABLE 1.4 ASMCMD Commands *(continued)*

View Name	In ASM Instance	In Database
V\$ASM_DISK	This view describes each disk that was discovered by the ASM instance. All disks are reported, even those not assigned to disk groups.	This view describes each disk that is assigned to a database. Note that discovery will occur each time you query this view. This can have performance impacts.
V\$ASM_DISKGROUP_STAT	This view is equivalent to the V\$ASM_DISKGROUP view.	This view contains a single row for each ASM disk group that is mounted by the local ASM instance. No discovery will occur when this view is queried.
V\$ASM_FILE	Displays each ASM file contained in the ASM instance.	Not used in a database instance.
V\$ASM_DISK_STAT	This view is equivalent to the V\$ASM_DISK view.	This view describes each disk that is assigned to a database. No discovery will occur when this view is queried.
V\$ASM_TEMPLATE	Displays each ASM template contained in the ASM instance by disk group.	Not used in a database instance.
V\$ASM_ALIAS	Displays each alias contained in the ASM instance by disk group.	Not used in a database instance.
V\$ASM_OPERATION	Displays each long-running operation occurring on the ASM instance.	Not used in a database instance.
V\$ASM_CLIENT	Displays each database that is using at least one disk group managed by the ASM instance.	Not used in a database instance.

Using ASM Storage

We have discussed management of an ASM instance. This section covers how to actually use ASM from an Oracle instance. You can put all sorts of Oracle-related files into an ASM instance, including these:

- Oracle datafiles
- Database tempfiles
- Online redo logs
- Archived redo logs
- Control files
- Spfiles
- RMAN backup sets
- The flash recovery area (FRA)
- Data-pump dump sets

When you create one of these objects, you can decide to create it in an ASM disk group. You can also define default file-creation locations that point to ASM disk groups. Finally, you can mix and match the use of ASM and cooked file systems if you prefer. In the following sections, we will review these topics:

- Defining what ASM files are
- Defining ASM as the default destination for database files
- Creating a tablespace using an ASM disk group as the destination
- Creating a control file in an ASM disk group location
- Creating spfiles or parameter files on an ASM disk group
- Creating online redo logs in an ASM disk group location
- Defining an ASM disk group location as an archived redo-log storage area
- Creating RMAN backup sets on an ASM disk
- Defining an ASM disk group as the location for the FRA

What Are ASM Files?

We have already created ASM disk groups. To actually use the ASM disk groups, we have to populate them with ASM files. In this section, we will discuss what ASM files are and then we will discuss the different kinds of ASM filenames that you might deal with.

ASM Files

ASM *files* are created in a number of different ways; for example, when you execute the `create tablespace` command and you indicate that the resulting datafile(s) should be stored in an ASM disk group, the result will be the creation of ASM files in that ASM disk group.

A goodly number of Oracle file types can be stored in ASM, including datafiles, control files, redo logs, and archived redo logs. There are some Oracle files that cannot be stored in an ASM group. These are mostly the administrative files like trace files, the alert log, and so on.

ASM Filename Types

When a file is created on an ASM disk, the filename is generated by ASM. There is a number of different kinds of ASM filename types:

- Fully qualified ASM filenames
- Numeric ASM filenames
- Alias ASM filenames
- Alias filenames with templates
- Incomplete filenames
- Incomplete filenames with templates

Let's look at each of these types in a bit more detail.

Fully Qualified ASM Filenames

The full filename is known as the *fully qualified ASM filename*. Here is an example of a fully qualified ASM filename:

```
+sp_dgroup2/mydb/controlfile/Current.56.544956473
```

The naming format for a fully qualified ASM filename is as follows:

- The `+group` is listed (in our case, `+sp_dgroup2`). Note that the `+` indicates the root of the ASM filename.
- The database name (in our case `mydb`).
- The file type (in our case, this is a control file).
- Next we have the start of the actual ASM file. First we have the file type flag (in our case, `Current`). This provides additional information on the file type in question. In this case, this is a current control file, as opposed to a control-file backup, which would be listed as `backup`.
- Finally we have two numbers delineated by a period (`56.544956473`), which represent the file number and an incarnation number. These two numbers combined guarantee that the ASM filename will be unique from any other ASM filename on the system.

ASM Numeric Filenames

The *ASM numeric filename* is a subset of the fully qualified filename, as you might have noticed. The numeric filename for the fully qualified filename in the preceding section would be

```
+sp_dgroup2.56.544956473
```

Alias ASM Filenames

An *ASM alias filename* takes on the following format:

```
+group_name/your_assigned_alias
```

In this case, if you assign the alias to the file when it's created filenames might look like this:

```
+sp_dgroup2/ctrl_files/control_file_01  
+sp_dgroup2/datafile/mydbf_user_data_01
```

Alias ASM Filenames with Templates

You can also reference an alias ASM filename with a template name. Simply include the ASM template name in parentheses next to the alias-name definition, as shown here:

```
+sp_dgroup2/ctrl_files/control_file_01(special)
```

Incomplete ASM Filenames

There are times you will use an *incomplete ASM filename*. An ASM filename is incomplete when the name of the ASM disk group is all that is referenced, as in this case:

```
+Sp_dgroup1
```

So, the incomplete ASM filename really is just a + followed by the disk group name. This is the most commonly used ASM filename type because this type is used when defining default destinations for database files (see more on that in the next section) in parameters, creating tablespaces, or performing RMAN database backups.

Incomplete ASM Filenames with Templates

As with alias filenames, you can also reference a template in an incomplete filename definition, as shown here:

```
+sp_dgroup1(special_template)
```

Adding ASM Filename Aliases to Existing Files

You can add filename aliases to ASM files that have already been created. To add the alias, use the `alter diskgroup` command with the `add alias` parameter. For example, if you wanted to create an alias for `ALIAS_TBS.260.613168611`, you would issue the following command:

```
Alter diskgroup cooked_dgroup1
add alias '+cooked_dgroup1/alias_dir/alias_tbs_01.dbf'
FOR '+cooked_dgroup1/11GDB/datafile/alias_tbs. 260.613168611';
```

Managing ASM File Alias Names

You can change ASM file alias names with the `rename alias` parameter of the `alter diskgroup` command, as shown in this example:

```
Alter diskgroup cooked_dgroup1
Rename alias '+cooked_dgroup1/alias_dir/alias_tbs_01.dbf'
To '+cooked_dgroup1/datafiles/alias_tbs_01.dbf';
```

You can use the `drop alias` command to drop ASM aliases, as in this example:

```
Drop alias '+cooked_dgroup1/datafiles/alias_tbs_01.dbf';
```

Drop Files from an ASM Disk Group

There may be cases where you will need to drop files from an ASM disk group (for example, the database is removed in an unordered fashion). To remove ASM files, use the `alter diskgroup` command with the `drop file` clause. Here is an example of removing a file from an ASM disk group (in this case, using an alias name):

```
alter diskgroup cooked_dgroup1
drop file '+cooked_dgroup1/alias_dir/alias_tbs_01.dbf';
```

Defining ASM as the Default Destination for Database Files

If you decide you want to allow Oracle to create all file types as ASM file types, you can set the values of various parameters such that ASM will automatically be employed. One of the big benefits of this feature is the standardization of your database, ensuring that all files get placed where they belong and in the ASM structure to which they belong. You can define

default ASM destinations by defining incomplete ASM filenames. The following database parameters take incomplete ASM filenames:

- DB_CREATE_FILE_DEST
- DB_CREATE_ONLINE_LOG_DEST_n
- DB_RECOVERY_FILE_DEST
- CONTROL_FILES
- LOG_ARCHIVE_DEST_n (log_archive_dest_format will be ignored)
- LOG_ARCHIVE_DEST (log_archive_dest_format will be ignored)
- STANDBY_ARCHIVE_DEST

Here is an example of using an incomplete name when setting the DB_CREATE_FILE_DEST parameter so that it will use the ASM disk group +sp_dgroup1:

```
alter system set db_create_file_dest='+cooked_dgroup1' scope=both;
```

Creating a Tablespace Using an ASM Disk Group as the Destination

There are different ways to create tablespaces using ASM disks. In this section, we will first look at creating an ASM tablespace, allowing the default ASM disk location to be used (as a result of having set the DB_CREATE_FILE_DEST parameter as we did earlier). We will then look at how to create a tablespace datafile by explicitly referencing the ASM disk group that it is supposed to be assigned to.

Creating Tablespaces Using Default ASM Assignments

Now that you have seen how to define a default ASM location, you can use the create tablespace command to create a tablespace that will have a file in the ASM disk group by default, as shown in this example:

```
create tablespace test_rgf datafile size 100k;
```

Let's see where Oracle put the datafile now by querying the DBA_DATA_FILES view:

```
Select tablespace_name, file_name
from dba_data_files Where tablespace_name='TEST_RGF';
```

```
TABLESPACE FILE_NAME
```

```
-----
TEST_RGF    +COOKED_DGROUP1/11gDB/datafile/test_rgf.256.613064385
```

Note in this example that Oracle went ahead and filled out the rest of the path, giving us a complete filename in the `DBA_DATA_FILES` view to work with. We can also see this new file in the ASM instance using the `V$ASM_FILES` view, as shown here:

```
SQL> select group_number, file_number, type, blocks, bytes from v$asm_file;
```

GROUP_NUMBER	FILE_NUMBER	TYPE	BLOCKS	BYTES
-----	-----	-----	-----	-----
1	256	DATAFILE	14	114688

If you want to drop a tablespace that contains ASM files, you need only issue the `drop tablespace` command. Oracle will clean up all of the ASM datafiles associated with that tablespace. You can have a mix of ASM datafiles and normal datafiles assigned to a tablespace, as shown in this `create table` statement:

```
Create tablespace part_asm_tbs
Datafile 'c:\oracle\oradata\11gdb\part_asm_tbs_01.dbf' size 10m,
'+COOKED_DGROUP1' size 100k;
```

Let's look and see where the datafiles were created:

```
Select tablespace_name, file_name
from dba_data_files Where tablespace_name='PART_ASM_TBS';
```

TABLESPACE_NAME	FILE_NAME
-----	-----
PART_ASM_TBS	C:\ORACLE\ORADATA\11GDB\PART_ASM_TBS_01.DBF
PART_ASM_TBS	+COOKED_DGROUP1/11GDB/datafile/part_asm_tbs.256.613066047

Note that in this case, if you drop the `PART_ASM_TBS` tablespace, only the ASM files related to that tablespace would be removed from the disk when you issue the `drop tablespace` command. In cases such as these, you need to make sure you include the `including contents and datafiles` parameter with the `drop tablespace` command.

Creating Tablespaces Referencing Specific ASM Disk Groups

There are going to be many times when you will not want to define a default ASM disk group to write all tablespaces to. In this case, you will want to reference the specific ASM disk group that you want a datafile created in when you issue the `create tablespace` command. Here is an example:

```
create tablespace another_test
datafile '+COOKED_DGROUP1' size 100k;
```

Let's see where Oracle put the datafile now by querying the DBA_DATA_FILES view:

```
select tablespace_name, file_name
from dba_data_files Where tablespace_name='ANOTHER_TEST';
```

```
TABLESPACE_NAME FILE_NAME
-----
ANOTHER_TEST      +COOKED_DGROUP1/11GDB/datafile/another_test.256.613065911
```

The `create tablespace` command comes with a number of different options when you are using cooked file systems, and there is no reason you cannot use those options when you are using ASM file systems. For example, you can create a tablespace with `autoextend` enabled, as shown here:

```
create tablespace another_test
datafile '+COOKED_DGROUP1' size 100k
autoextend on next 10m maxsize 300m;
```

If you want to create a tablespace using a template other than the default template, this is easy too, as shown in this example:

```
create tablespace different_template
datafile '+COOKED_DGROUP1(alternate_template)';
```

Tablespace Maintenance When Using Tablespaces Referencing Specific ASM Disk Groups

Tablespace maintenance is basically unchanged when using ASM disks. For example, you can add a datafile with the `alter tablespace` command as you normally would:

```
alter tablespace part_asm_tbs Add datafile '+COOKED_DGROUP1' size 100k;
```

Creating a Database Using ASM Disk Group Locations

You can create a database that completely uses ASM disk group locations. You can do this when creating the database through the DBCA interface or if you are going to manually create the database. In the following sections, we will look at both options.

Creating a Database Using ASM Disks with DBCA

If you are creating the database with the DBCA, you will have the opportunity to indicate that you want to use ASM disks for the database as a part of the DBCA workflow.

After you indicate that you want to use ASM, you will be prompted to create the SYS password to the ASM instance. DBCA will then present to you a list of available disk groups that you can use to create the database.

DBCA will confirm your selection in the next screen. You will then be presented with a screen that asks you where you want the flash recovery area to be assigned. It will prepopulate this screen with one of the disk groups that you selected to use in the previous screen (typically it will be the second disk group in the list).

The DBCA will set all of the different file location parameters to those in the ASM disk groups that you selected (for example, `CONTROL_FILES`, `DB_CREATE_FILE_DEST`, `DB_RECOVERY_FILE_DEST`, and `LOG_ARCHIVE_DEST_1`). You can edit these choices if you want to use a mix of ASM and cooked file systems for some reason.

You complete the DBCA screens as normal and DBCA will then create a database that is totally (or partially, if you prefer) using ASM.

Creating a Database Manually Using ASM Disks

It can actually be easier to create a database using ASM than to create a database with DBCA, if only because less typing is required! To create an Oracle database with ASM, follow the standard procedures, but when you are creating the parameter file, make sure you assign the following parameters to an ASM disk group:

- `DB_CREATE_FILE_DEST`
- `DB_RECOVERY_FILE_DEST`

Once you have done this, issuing the `create database` command requires no parameters at all! Simply issue the command and that's it! Oracle will create the various database files on ASM disk.

Creating a Control File in an ASM Disk Group Location

When you create a database from the DBCA or manually, you can opt to create the database control files in an ASM location by setting the `CONTROL_FILES` parameter to an ASM disk group. A note about moving control files, or any other database-related files: be very careful. It's dangerous to re-create the control file because it is a rather central part of your database!

If you have an existing database and you want to move the control files to ASM, it gets a bit stickier. You will pretty much have to use the `create controlfile` command to move the database control files to ASM disks. You will need to change the database `CONTROL_FILES` parameter before you run the `create controlfile` command. Here is an example of this operation:

```
SQL> alter system set control_files='' scope=SPFILE;
SQL> alter system set
DB_CREATE_FILE_DEST='+COOKED_DGROUP1','+COOKED_DGROUP2' scope=spfile;
SQL> shutdown immediate
SQL> startup nomount
SQL>CREATE CONTROLFILE REUSE DATABASE "11GDB"
```

```

NORESETLOGS NOARCHIVELOG
  MAXLOGFILES 16
  MAXLOGMEMBERS 3
  MAXDATAFILES 100
  MAXINSTANCES 8
  MAXLOGHISTORY 292
LOGFILE
  GROUP 1 'C:\ORACLE\ORADATA\11GDB\REDO01.LOG' SIZE 50M,
  GROUP 2 'C:\ORACLE\ORADATA\11GDB\REDO02.LOG' SIZE 50M,
  GROUP 3 'C:\ORACLE\ORADATA\11GDB\REDO03.LOG' SIZE 50M
-- STANDBY LOGFILE
DATAFILE
  'C:\ORACLE\ORADATA\11GDB\SYSTEM01.DBF',
  'C:\ORACLE\ORADATA\11GDB\UNDOTBS01.DBF',
  'C:\ORACLE\ORADATA\11GDB\SYSAUX01.DBF',
  'C:\ORACLE\ORADATA\11GDB\USERS01.DBF',
  'C:\ORACLE\ORADATA\11GDB\EXAMPLE01.DBF',
  'C:\ORACLE\ORADATA\11GDB\PART_ASM_TBS_01.DBF',
  '+COOKED_DGROUP1/11GDB/datafile/part_asm_tbs.256.613066047',
  '+COOKED_DGROUP1/11GDB/datafile/part_asm_tbs.257.613083267'
CHARACTER SET WE8MSWIN1252;
SQL>RECOVER DATABASE;
SQL>ALTER DATABASE OPEN NORESETLOGS;

```

You can also use RMAN to restore the control file to an ASM disk location, as shown in this example (this assumes you are connected to a recovery catalog):

```

SQL> alter system set control_files='' scope=SPFILE;
SQL> alter system set
DB_CREATE_FILE_DEST='+COOKED_DGROUP1' scope=spfile;
RMAN>shutdown
RMAN>startup nomount
RMAN>restore controlfile;
RMAN>recover database;
RMAN>alter database open resetlogs;

```

If you are using autobackups, then the process is slightly different:

```

SQL> alter system set control_files='' scope=SPFILE;
SQL> alter system set
DB_CREATE_FILE_DEST='+COOKED_DGROUP1' scope=spfile;

```

```

RMAN>shutdown
RMAN>startup nomount
RMAN>restore controlfile from autobackup;
RMAN>recover database;
RMAN>alter database open resetlogs;

```

Note that instead of setting the `DB_CREATE_FILE_DEST` parameter, you could set the `CONTROL_FILES` parameter, as shown here:

```

alter system set control_files=
'+COOKED_DGROUP1/11GDB/controlfile/current.259.613088323',
'+COOKED_DGROUP2/11GDB/controlfile/current.257.613088325';

```

If you have restored the control files to a disk group, you may not know what the file-names are in order to set the `CONTROL_FILES` parameter (the RMAN output does not give you the ASM filenames that are created). In this case, you can query the `V$ASM_FILE` view from the ASM instance and derive the filename for the newly created control file.

Recall that the format for an ASM filename is `file_type_flag.file#.incarnation#`. Knowing this, you can derive the control-file names. For example, here is a query against the `V$ASM_FILE` view in our database after we restored the control files to our ASM instance:

```

SQL> select a.group_number, b.name, a.incarnation, a.file_number, a.type
2   from v$asm_diskgroups b, v$asm_file a
3  where a.group_number= b.group_number
4  and a.type='CONTROLFILE';

```

GROUP_NUMBER	NAME	INCARNATION	FILE_NUMBER	TYPE
1	COOKED_DGROUP1	613087119	258	CONTROLFILE
2	COOKED_DGROUP2	613087131	256	CONTROLFILE

From this, we can surmise that our `CONTROL_FILES` parameter should be set to the following:

```

alter system set control_files=
'+COOKED_DGROUP1/11GDB/controlfile/current.258.613087119',
'+COOKED_DGROUP1/11GDB/controlfile/current.258.613087131'
Scope=spfile;

```

Of course, you can reverse this process and move control files out of ASM and put them into cooked file systems at any time. As should always be standard practice, make sure you back up your database before you begin moving any database-related files around.

Creating Spfiles or Parameter Files on an ASM Disk Group

You can create pfiles or spfiles on an ASM disk group using the `create pfile` or `create spfile` command with the ASM disk group as the location for the parameter file. For example, you could issue this command:

```
create spfile '+COOKED_DGROUP1' from pfile;
```

Creating Online Redo Logs in an ASM Disk Group Location

Creation of redo logs on ASM disks is straightforward. If the database was configured to use ASM from the beginning, then the existing online redo logs will already be on ASM disk groups. Assuming that the parameter `DB_ONLINE_CREATE_LOG_DEST_n` or `DB_CREATE_FILE_DEST` is set, you can simply issue the `alter database add logfile` command and Oracle will add a new redo log group to your database for you, as shown in this example:

```
alter database add logfile size 100m;
```

You can also manually add a redo log file group to a disk group if you prefer using SQL, such as in the following, which will create a new log file group, and multiplex it, between two ASM disk groups:

```
alter database add logfile ('+COOKED_DGROUP1', '+COOKED_DGROUP2') size 100m;
```

Defining an ASM Disk Group Location as an Archived Redo Log Storage Area

Once you have created an ASM disk, it's easy to use it as the storage for archived redo logs. Simply set one of the `LOG_ARCHIVE_DEST_n` parameters to point to that ASM disk group, as shown in this example:

```
alter system  
set log_archive_dest_1='location=+COOKED_DGROUP2';
```

You can check the ASM instance and see the new archived logs being created, as shown in this example code:

```
SQL> select a.group_number, b.name, a.incarnation, a.file_number, a.type  
2   from v$ASM_DISKGROUPS b, v$asm_file a  
3   where a.group_number=b.group_number  
4   and a.type='ARCHIVELOG';
```

GROUP_NUMBER	NAME	INCARNATION	FILE_NUMBER	TYPE
2	COOKED_DGROUP2	613091705	258	ARCHIVELOG

Creating Database Objects Using ASM Filename Aliases

An alias includes the disk group name and then appends a user-defined name to the filename. This makes it possible to reference an ASM file with a name that makes some sense. ASM aliases start with the disk group name, followed by a slash and then the alias name as in this example:

```
+COOKED_DGROUP1/datafiles/myfile.dbf
```

Aliases can be created at the time the file is created (such as when you issue the `create tablespace` command), or you can add the alias later. Here is an example of creating a tablespace with an alias filename:

```
create tablespace alias_tbs
Datafile'+COOKED_DGROUP1/myalias.dbf' size 10m;
```

The resulting ASM file would be as follows:

```
+COOKED_DGROUP1/11GDB/datafile/alias_tbs.256.613066047
```

You can see this through the following query executed in the ASM instance:

```
SQL> select b.name gname, a.name aname, a.system_created, a.alias_directory,
2         c.type file_type
3         from v$asm_alias a, v$ASM_DISKGROUPS b, v$asm_file c
4         where a.group_number = b.group_number
5         and a.group_number = c.group_number(+)
6         and a.file_number = c.file_number(+)
7         and a.file_incarnation = c.incarnation(+)
8         and b.name='COOKED_DGROUP1'
9         and c.type='DATAFILE';
```

GNAME	ANAME	S	A	FILE_TYPE
COOKED_DGROUP1	ALIAS_TBS.260.613168611	Y	N	DATAFILE

Using RMAN with ASM

You can use RMAN in conjunction with ASM. In the following sections, we will cover the following RMAN-related operations:

- Copying database datafiles to an ASM disk with RMAN
- Creating RMAN backups on ASM

Copying Database Datafiles to an ASM Disk with RMAN

If you want to move your entire database to ASM, you can easily do this with RMAN. First you make an image copy of the database, copying it to an ASM disk group. Then use the RMAN `switch database to copy` command to switch the database from using the old datafiles to using the new datafiles that were copied onto the ASM drives. Here is an example of moving the database datafiles using the following commands:

```
RMAN>shutdown
RMAN>startup mount
RMAN>backup as copy database format '+COOKED_DGROUP1';
RMAN>switch database to copy;
RMAN>alter database open;
```

Creating RMAN Backups on ASM

RMAN backup sets can be made to ASM disks. This means that the database, archived redo logs, control-file backups, and spfiles can all be backed up to ASM disks. There are two different ways of using ASM for backups. You can send individual backups directly to an ASM disk group, or you can define the flash recovery area to be a disk group and cause backups to be sent to the flash recovery area. Let's look at these two options in a bit more detail.

Backing Up from RMAN to ASM Directly

Backing up to an ASM disk group with RMAN is quite easy. Use the RMAN `backup` command and add the `format` parameter indicating the disk group to which you want to back up the database. Here is an example:

```
RMAN>backup as compressed backupset database format '+COOKED_DGROUP1';
```

You can also back up archived redo logs and database control files using RMAN via the same method.

Configuring and Backing Up to an ASM Flash Recovery Area

The flash recovery area is a directory structure that centralizes Oracle backups in one Oracle-defined backup structure (see Chapter 2 for more on the flash recovery area). You define the flash recovery area by setting the `DB_RECOVERY_FILE_DEST` and `DB_RECOVERY_FILE_DEST_SIZE` parameters as required. Here is an example of using the `alter system` command to point the flash recovery area to an ASM disk group.

```
alter system set db_recovery_file_dest='+COOKED_DGROUP1';  
alter system set db_recovery_file_dest_size=4G;
```

Once these parameters have been set, RMAN backups will start using the flash recovery area and ASM since the flash recovery area has been configured to use an ASM disk group.

Summary

In this chapter, we showed you how Automatic Storage Management (ASM) can reduce or eliminate the headaches involved in managing the disk space for all Oracle file types, including online and archived logs, RMAN backup sets, flashback logs, and even initialization parameter files (spfiles).

We reviewed the concepts related to a special type of instance called an ASM instance along with the initialization parameters specific to an ASM instance. In addition, we described the dynamic performance views that allow you to view the components of an ASM disk group as well as to monitor the online rebalancing operations that occur when disks are added to or removed from a disk group. Starting and stopping an ASM instance is similar to starting and stopping a traditional database instance. Of course, other databases that use disk groups within the ASM instance will not be available to users if the ASM instance is not available to service disk group requests.

ASM filenames have a number of different formats and are used differently depending on whether existing ASM files or new ASM files are being referenced. ASM templates are used in conjunction with ASM filenames to ease the administration of ASM files.

Additionally, we reviewed ASM disk group architecture, showing how failure groups can provide redundancy and performance benefits while eliminating the need for a third-party logical volume manager. Dynamic disk group rebalancing automatically tunes I/O performance when a disk is added or deleted from a disk group or a disk in a disk group fails.

Exam Essentials

Enumerate the benefits and characteristics of Automatic Storage Management (ASM).

Understand how ASM can relieve you of manually optimizing I/O across all files in the tablespace by using ASM disk groups. Show how ASM operations can be performed online with minimal impact to ongoing database transactions.

Be able to create an ASM instance and configure its initialization parameters. Understand the initialization parameters `INSTANCE_TYPE`, `ASM_POWER_LIMIT`, `ASM_DISKSTRING`, and `ASM_DISKGROUPS`. Configure `DB_UNIQUE_NAME` for an ASM instance. Start up and shut down an ASM instance, noting the dependencies with database instances that are using the ASM instance's disk groups.

Understand the architecture of an ASM instance. Enumerate the different states for an ASM instance. Describe what happens when an ASM instance is shut down normally or is aborted. Understand and describe the differences between an RDBMS instance and an ASM instance.

Understand redundancy and resync. Describe what redundancy is and how it's implemented in ASM. Understand what a failure group is and how it is created. Know what ASM fast disk resync and preferred mirror read are.

Understand how ASM filenames are constructed and used when creating Oracle objects. Differentiate how different ASM filename formats are used and how files are created depending on whether the file is an existing ASM file, whether a new ASM file is being created, or whether multiple ASM files are being created. Understand the different system templates for creating ASM files with the associated filename and how the characteristics are applied to the ASM files. Show how ASM files are used in SQL commands.

Be able to create, drop, and alter ASM disk groups. Define multiple failure groups for new disk groups and make sure you understand how the number of failure groups is different for two-way and three-way mirroring. Show how disk rebalancing can be controlled or rolled back. Understand the ASM disk group attributes and how they are used.

Identify the steps involved in converting non-ASM files to ASM files using RMAN. Migrate a database to ASM disk groups by shutting down the database, running an RMAN script for each file to be converted, and opening the database with `RESETLOGS`.

Review Questions

1. What are three benefits of using ASM? (Choose three.)
 - A. Ease of disk administration and maintenance
 - B. Load balancing across physical disks
 - C. Software RAID-1 data redundancy with double or triple mirrors
 - D. Automatic recovery of failed disks
2. What components are present in an ASM instance? (Choose three.)
 - A. SGA
 - B. Database processes
 - C. Database datafiles
 - D. Control files
 - E. Database parameter file or spfile
3. Which of the following is a benefit of ASM fast disk resync?
 - A. Failed disks are taken offline immediately but are not dropped.
 - B. Disk data is never lost.
 - C. By default, the failed disk is not dropped from the disk group ever, protecting you from loss of that disk.
 - D. The failed disk is automatically reformatted and then resynchronized to speed up the recovery process.
 - E. Hot spare disks are automatically configured and added to the disk group.
4. What is the result of increasing the value of the parameter `ASM_POWER_LIMIT` during a rebalance operation?
 - A. The ASM rebalance operation will likely consume fewer resources and complete in a shorter amount of time.
 - B. The ASM rebalance operation will consume fewer resources and complete in a longer amount of time.
 - C. The ASM rebalance operation will be parallelized and should complete in a shorter amount of time.
 - D. There is no `ASM_POWER_LIMIT` setting used in ASM.
 - E. None of the above.

5. What is the default AU size of an ASM disk group? What is the maximum AU size in an ASM disk group?
- A. 100KB default, 10TB maximum
 - B. 256KB default, 1024MB maximum
 - C. 10MB default, 126PB maximum
 - D. 64KB default, 1EB maximum
 - E. 1MB default, 64MB maximum
6. Which initialization parameter in an ASM instance specifies the disk groups to be automatically mounted at instance startup?
- A. ASM_DISKMOUNT
 - B. ASM_DISKGROUP
 - C. ASM_DISKSTRING
 - D. ASM_MOUNTGROUP
7. When an ASM instance receives a SHUTDOWN NORMAL command, what command does it pass on to all database instances that rely on the ASM instance's disk groups?
- A. TRANSACTIONAL
 - B. IMMEDIATE
 - C. ABORT
 - D. NORMAL
 - E. None of the above
8. When starting up your ASM instance, you receive the following error:
- ```
SQL> startup pfile=?/dbs/init+ASM.ora
ASM instance started
Total System Global Area 104611840 bytes
Fixed Size 1298220 bytes
Variable Size 78147796 bytes
ASM Cache 25165824 bytes
ORA-15032: not all alterations performed
ORA-15063: ASM discovered an insufficient number of disks for disk group
"DGROUP3"
ORA-15063: ASM discovered an insufficient number of disks for disk group
"DGROUP2"
ORA-15063: ASM discovered an insufficient number of disks for disk group
"DGROUP1"
```

In trying to determine the cause of the problem, you issue this query:

```
SQL> show parameter asm
```

| NAME                              | TYPE    | VALUE                     |
|-----------------------------------|---------|---------------------------|
| asm_allow_only_raw_disks          | boolean | FALSE                     |
| asm_diskgroups                    | string  | DGROUP1, DGROUP2, DGROUP3 |
| asm_diskstring                    | string  |                           |
| asm_power_limit                   | integer | 1                         |
| asm_preferred_read_failure_groups | string  |                           |

What is the cause of the error?

- A. The ASM\_DISKGROUPS parameter is configured for three disk groups: DGROUP1, DGROUP2, and DGROUP3. The underlying disks for these disk groups have apparently been lost.
  - B. The format of the ASM\_DISKGROUPS parameter is incorrect. It should reference the disk group numbers, not the names of the disk groups.
  - C. The ASM\_POWER\_LIMIT parameter is incorrectly set to 1. It should be set to the number of disk groups being attached to the ASM instance.
  - D. The ASM\_DISKSTRING parameter is not set; therefore disk discovery is not possible.
  - E. There is insufficient information to solve this problem.
9. As DBA for the Rebel Alliance you have decided that you need to facilitate some redundancy in your database. Using ASM, you want to create a disk group that will provide for the greatest amount of redundancy for your ASM data (you do not have advanced SAN mirroring technology available to you, unfortunately). Which of the following commands would create a disk group that would offer the maximum in data redundancy?
- A. 

```
CREATE DISKGROUP dg_alliance1 NORMAL REDUNDANCY
failgroup diskcontrol1 DISK 'c:\oracle\asm_disk_file_disk3' NAME file_
diska1
failgroup diskcontrol2 DISK 'c:\oracle\asm_disk_file_disk4' NAME file_
diskb1;
```
  - B. 

```
CREATE DISKGROUP dg_alliance1 EXTERNAL REDUNDANCY
failgroup diskcontrol1 DISK 'c:\oracle\asm_disk_file_disk3' NAME file_
diska1;
```
  - C. 

```
CREATE DISKGROUP dg_alliance1
HIGH REDUNDANCY
failgroup diskcontrol1 DISK 'c:\oracle\asm_disk_file_disk1' NAME file_
disk1
failgroup diskcontrol2 DISK 'c:\oracle\asm_disk_file_disk2' NAME file_
disk2
failgroup diskcontrol2 DISK 'c:\oracle\asm_disk_file_disk3' NAME file_
disk3;
```

- D. `CREATE DISKGROUP dg_alliance1  
MAXIMUM REDUNDANCY  
failgroup diskcontrol1 DISK 'c:\oracle\asm_disk\_file_disk1' NAME file_  
disk1  
failgroup diskcontrol2 DISK 'c:\oracle\asm_disk\_file_disk2' NAME file_  
disk2  
failgroup diskcontrol2 DISK 'c:\oracle\asm_disk\_file_disk3' NAME file_  
disk3  
failgroup diskcontrol2 DISK 'c:\oracle\asm_disk\_file_disk4' NAME file_  
disk4;`
- E. None of the above
10. You want to migrate your database to ASM, so you've done a clean shutdown, made a closed backup of the entire database, noted the location of your control files and online redo log files, and changed your spfile to use OMF. The last step is to run an RMAN script to do the conversion. Using the following steps, what is the correct order in which the following RMAN commands should be executed?
1. `STARTUP NOMOUNT`
  2. `ALTER DATABASE OPEN RESETLOGS`
  3. `SQL "ALTER DATABASE RENAME 'logfile1 path' TO '+dgrp4 '"` # plus all other logfiles
  4. `SWITCH DATABASE TO COPY`
  5. `BACKUP AS COPY DATABASE FORMAT '+dgrp4'`
  6. `ALTER DATABASE MOUNT`
  7. `RESTORE CONTROLFILE FROM 'controlfile_location'`
- A. 2, 5, 3, 1, 7, 6, 4  
B. 1, 7, 6, 5, 4, 3, 2  
C. 5, 1, 2, 7, 4, 6, 3  
D. 7, 3, 1, 5, 6, 2, 4
11. How can you reverse the effects of an `ALTER DISKGROUP ... DROP DISK` command if it has not yet completed?
- A. Issue the `ALTER DISKGROUP ... ADD DISK` command.
  - B. Issue the `ALTER DISKGROUP ... UNDROP DISKS` command.
  - C. Issue the `ALTER DISKGROUP ... DROP DISK CANCEL` command.
  - D. Retrieve the disk from the Recycle Bin after the operation completes.

12. To reference existing ASM files, you need to use a fully qualified ASM filename. Your development database has a disk group named DG2A, the database name is DEV19, and the ASM file that you want to reference is a datafile for the USERS02 tablespace. Which of the following is a valid ASM filename for this ASM file?
- A. dev19/+DG2A/datafile/users02.701.2
  - B. +DG2A/dev19/datafile/users02.701.2
  - C. +DG2A/dev19/users02/datafile.701.2
  - D. +DG2A.701.2
  - E. +DG2A/datafile/dev19.users.02.701.2
13. Which background process coordinates the rebalance activity for disk groups?
- A. ORBn
  - B. OSMB
  - C. RBAL
  - D. ASMn
14. On the development database rac0, there are six raw devices: /dev/raw/raw1 through /dev/raw/raw6. /dev/raw/raw1 and /dev/raw/raw2 are 8GB each, and the rest are 6GB each. An existing disk group +DATA1, of NORMAL REDUNDANCY, uses /dev/raw/raw1 and /dev/raw/raw2. Which series of the following commands will drop one of the failure groups for +DATA1, create a new disk group +DATA2 using two of the remaining four raw devices, and then cancel the drop operation from +DATA1?
- A. ALTER DISKGROUP DATA1 DROP DISK DATA1\_0001;CREATE DISKGROUP DATA2 NORMAL REDUNDANCY FAILGROUP DATA1A DISK '/dev/raw/raw3' FAILGROUP DATA1B DISK '/dev/raw/raw4';ALTER DISKGROUP DATA1 UNDROP DISKS;
  - B. ALTER DISKGROUP DATA1 DROP DISK DATA1\_0001;CREATE DISKGROUP DATA2 HIGH REDUNDANCY FAILGROUP DATA1A DISK '/dev/raw/raw3' FAILGROUP DATA1B DISK '/dev/raw/raw4';ALTER DISKGROUP DATA1 UNDROP DISKS;
  - C. ALTER DISKGROUP DATA1 DROP DISK DATA1\_0001;CREATE DISKGROUP DATA2 NORMAL REDUNDANCY FAILGROUP DATA1A DISK '/dev/raw/raw3' FAILGROUP DATA1B DISK '/dev/raw/raw4';ALTER DISKGROUP DATA1 UNDROP DATA1\_0001;
  - D. ALTER DISKGROUP DATA1 DROP DISK DATA1\_0001 ADD DISK GROUP DATA2 NORMAL REDUNDANCY FAILGROUP DATA1A DISK '/dev/raw/raw3' FAILGROUP DATA1B DISK '/dev/raw/raw4';ALTER DISKGROUP DATA1 UNDROP DISKS;
15. Which type of database file is spread across all disks in a disk group?
- A. All types of files are spread across all disks in the disk group.
  - B. Datafiles
  - C. Redo log files
  - D. Archived redo log files
  - E. Control files

16. How can you reverse the effects of an ALTER DISKGROUP ... DROP DISK command if it has already completed?
- A. Issue the ALTER DISKGROUP ... ADD DISK command.
  - B. Issue the ALTER DISKGROUP ... UNDROP DISKS command.
  - C. Issue the ALTER DISKGROUP ... DROP DISK CANCEL command.
  - D. Retrieve the disk from the Recycle Bin after the operation completes.
17. Which of the following ALTER DISKGROUP commands does *not* use V\$ASM\_OPERATION to record the status of the operation?
- A. ADD DIRECTORY
  - B. DROP DISK
  - C. RESIZE DISK
  - D. REBALANCE
  - E. ADD FAILGROUP
18. If you use ALTER DISKGROUP ... ADD DISK and specify a wildcard for the discovery string, what happens to disks that are already a part of the same or another disk group?
- A. The command fails unless you specify the FORCE option.
  - B. The command fails unless you specify the REUSE option.
  - C. The command must be reissued with a more specific discovery string.
  - D. The other disks, already part of the disk group, are ignored.
19. You are an Oracle DBA responsible for an ASM instance. The disk controller on your system fails. You suspect that the disk itself is okay. You know it will take 24 hours to replace the controller and you don't want to have to rebuild the disks from scratch. What do you do?
- A. Take the whole disk group offline and wait for the controller card to be installed. Once it's installed, bring the disk group online again.
  - B. Change the ASM parameter ASM\_PREFERRED\_READ\_FAILURE\_GROUPS to indicate that you want to read from the non-failed disk. Once the disk controller is replaced, reset the parameter to its original value.
  - C. You have no choice but to rebuild the disk. Drop the disk from the disk group and wait for the controller to be replaced. Once the controller is replaced, add the disk back into the disk group and allow ASM to rebuild it.
  - D. If you are using any setting other than REDUNDANCY EXTERNAL for your disk group, you will have to recover any data on that disk from a backup. The database will be unavailable until you can correct the problem and perform recovery.
  - E. Change the attribute DISK\_REPAIR\_TIME on the disk group to a time greater than 24 hours.

- 20.** As the DBA, you run the following query on your ASM instance. What is the implication of the results of the query? (Choose two.)

```
SQL> select group_number, name, state from v$ASM_DISKGROUP;
```

| GROUP_NUMBER | NAME    | STATE      |
|--------------|---------|------------|
| 0            | DGROUP1 | DISMOUNTED |
| 2            | DGROUP2 | MOUNTED    |
| 3            | DGROUP3 | MOUNTED    |

- A.** The DGROUP1 disk group was unmounted by another DBA.
- B.** A datafile has been lost, causing the ASM disk group DGROUP1 to go into the DISMOUNTED state.
- C.** One of the redundant disks (DGROUP1) has been lost in a disk group.
- D.** This query has no meaning in an ASM instance.
- E.** A disk associated with a disk group was discovered after the ASM instance initially opened.

# Answers to Review Questions

1. A, B, C. Option A is correct because ASM makes administration and maintenance of disks much easier. Option B is correct because ASM provides for load balancing across the physical disks for better performance. Answer C is correct because ASM provides RAID-1 redundancy via double or triple mirrors.
2. A, B, E. Option A is correct because the SGA is allocated when the ASM instance is started. Option B is correct because the Oracle processes are also started when the ASM instance is started. Option E is correct because the ASM instance requires either a parameter file or an spfile.
3. A. When a disk fails, it is taken offline immediately but it is not dropped at that time. By default, you will have 3.6 hours to correct the problem before ASM will automatically drop the disk. You can configure this time by modifying the disk attributes.
4. C. Increasing the value of `ASM_POWER_LIMIT` will increase the degree of parallelism of the rebalance operation, which may help to increase the performance of that operation.
5. E. The default AU size is 1MB, and 64MB is the maximum AU size for a disk group.
6. B. The initialization parameter `ASM_DISKGROUP`, valid only in an ASM instance, specifies the disk groups to be automatically mounted when the ASM instance starts. `ASM_DISKSTRING` is operating system-dependent and restricts the file-system devices that can be used to create disk groups. `ASM_DISKMOUNT` and `ASM_MOUNTGROUP` are not valid initialization parameters.
7. E. If you do a normal shutdown of the ASM instance, an error will be returned if any Oracle database is using that ASM instance. Use the `shutdown abort` command to force the ASM instance to shut down. This will cause all other Oracle databases attached to the ASM instance to be shut down with the equivalent of a `shutdown abort` command.
8. D. The `ASM_DISKSTRING` parameter is not set correctly. When the ASM instance is started, it will use the `ASM_DISKSTRING` to do ASM disk discovery. Correct the `ASM_DISKSTRING` parameter, and restart the instance to correct the problem.
9. C. High redundancy is the highest redundancy setting available in ASM, or to the Alliance in this case. This will result in a double-mirrored ASM disk group.
10. B. After the `RMAN` script is run and the database is up and running successfully, you may delete the old database files.
11. B. If the `DROP DISK` operation has not yet completed, you can cancel and roll back the entire `DROP DISK` operation by using `ALTER DISKGROUP ... UNDROP DISKS`, with the disk group still being continuously available to all users.
12. B. A fully qualified existing ASM filename has the format `+group/dbname/filetype/tag.file.incarnation`. In this case, *filetype* is *datafile*, and *tag* is the tablespace name to which it belongs, or `users02`.

- 13. C. RBAL coordinates rebalance activity for a disk group in an ASM instance.
- 14. A. Note that the UNDROP operation will cancel a drop operation in progress but cannot reverse a drop operation that has already completed. For HIGH REDUNDANCY, at least three failure groups must be specified. While you can combine a drop and add operation into one command, the command can reference only one disk group.
- 15. A. All types of database files are spread across all disks in the disk group to ensure redundancy unless the redundancy is set to EXTERNAL.
- 16. A. If the DROP DISK operation has already completed, you must use ALTER DISKGROUP ... ADD DISK to add the disk back to the disk group. In any case, the disk group is continuously available to all users and no data is lost.
- 17. A. The ADD DIRECTORY command is not likely to use V\$ASM\_OPERATION to track its progress, because this operation adds only a small amount of metadata—a directory object—to the disk group and takes a minimal amount of time to complete. The V\$ASM\_OPERATION view provides the status of long-running ASM operations.
- 18. D. The ALTER DISKGROUP ... ADD DISK command adds all disks that match the discovery string but are not already part of the same or another disk group.
- 19. E. The DISK\_REPAIR\_TIME attribute will prevent Oracle from automatically dropping the disk in the disk group for a specific period of time. This gives you time to replace the controller indicated in the question.
- 20. A, E. Apparently, for some reason DGROUP1 was not mounted when the ASM instance was started, or the disk was missing and then reappeared (hardware failure perhaps) after the ASM instance was started. ASM will discover new disks, even after the ASM instance is opened.